Distributed Predictive Coding for Spatio-Temporally Correlated Sources

Ankur Saxena, *Member, IEEE*, and Kenneth Rose, *Fellow, IEEE*

***Abstract*—Distributed coding of correlated sources with memory poses a number of considerable challenges that threaten its practical application, particularly (but not only) in the context of sensor networks. This problem is strongly motivated by the obvious observation that most common sources exhibit temporal correlations that may be at least as important as spatial or intersource correlations. This paper presents an analysis of the underlying tradeoffs, paradigms for coding systems, and approaches for distributed predictive coder design optimization. Motivated by practical limitations on both complexity and delay (especially for dense sensor networks) the focus here is on predictive coding. From the source coding perspective, the most basic tradeoff (and difficulty) is due to conflicts that arise between distributed coding and prediction, wherein "standard" distributed quantization of the prediction errors, if coupled with imposition of zero decoder drift, would drastically compromise the predictor performance and hence the ability to exploit temporal correlations. Another challenge arises from instabilities in the design of closed-loop predictors, whose impact has been observed in the past, but is greatly exacerbated in the case of distributed coding. In the distributed predictive coder design, we highlight the fundamental tradeoffs encountered within a more general paradigm where decoder drift is allowable or unavoidable, and must be effectively accounted for and controlled. We derive an overall design optimization method for distributed predictive coding that avoids the pitfalls of naive distributed predictive quantization and produces an optimized low complexity and low delay coding system. The proposed iterative algorithms for distributed predictive coding subsume traditional single-source predictive coding and memoryless distributed coding as extreme special cases.**

***Index Terms*—Distributed quantization, predictive coding, sensor networks.**

I. Introduction

THE theoretical foundation of the field of distributed source coding (DSC) was laid in the early seventies with the seminal work of Slepian and Wolf [1], which was followed shortly afterwards by Wyner and Ziv in [2]. A considerable revival of interest, with focus on practical code design, has been witnessed since the late nineties, with the work of Pradhan and Ramchandran [3] as a notable precursor. The field eventually saw the emergence of various distributed coding techniques, mostly with an eye towards sensor networks (see reviews in [4] and [5]). The basic setting in DSC involves multiple correlated sources (e.g., data collected by spatially distributed sensors) whose information is sent to a fusion center without any intercommunication amongst transmitters (see Fig. 1). The main objective in DSC is to exploit intersource correlations despite the fact that each sensor source is encoded without access to other sources. The only information available to a source encoder about other sources involves their joint statistics (possibly extracted from training data).

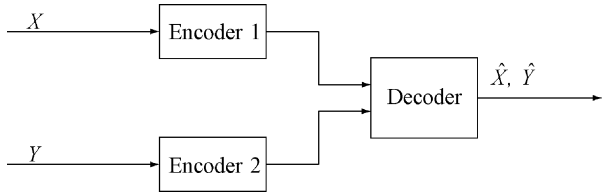

Fig. 1. Distributed coding of two correlated sources.

The main motivation for the work presented here springs from the fact that most sources in the real-world are sources with memory, i.e., they exhibit temporal correlations. In particular, sensor networks will often produce data whose time correlations are at least as important as their spatial (intersource) correlations. Examples range from simple sensors monitoring slowly varying physical quantities such as temperature or pressure, to the extreme of video cameras collecting highly correlated frame sequences.

Realizing the prevalence of sources with memory and the importance of exploiting both temporal and intersource correlation, we reformulate the problem within the representative setting of distributed *predictive* coding (DPC) systems. Given the historical focus on intersource correlations, most existing DSC work naturally addressed memoryless sources where one need not worry about temporal correlations. The implicit assumption may have been that predictive coding per se is a largely solved problem, and that extending DSC results to incorporate prediction would require a straightforward integration phase. (An alternative argument may involve handling long blocks of source data, as in vector quantization to exploit time correlations, but the cost in delay and complexity may be considerable). We shall, however, see that the generalization from DSC to DPC is highly nontrivial due to conflicting objectives of distributed coding versus efficient prediction in DPC. In other words, optimal distributed coding (in terms of current reconstruction quality) may severely compromise the prediction loop at each source encoder. We therefore propose

Manuscript received August 13, 2008; accepted April 23, 2009. First published June 02, 2009; current version published September 16, 2009. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Christine Guillemot. The work was supported in part by the NSF under grants IIS-0329267 and CCF-0728986, the University of California MICRO program, Applied Signal Technology Inc., Cisco Systems Inc., Dolby Laboratories Inc., Qualcomm Inc., and Sony Ericsson, Inc.. The material in this paper was presented in part at the IEEE International Symposium on Information Theory, June 2007, and the IEEE Information Theory Workshop, September 2007.

The authors are with the Department of Electrical & Computer Engineering, University of California, Santa Barbara, CA 93106 USA (e-mail: ankur@ece.ucsb.edu; rose@ece.ucsb.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSP.2009.2024258

to investigate new DPC system paradigms and methods to optimize their design.

There exists a channel coding "camp" of DSC research (see [6] and [7]), where long delays may be employed to achieve the desired performance, (e.g., using turbo/LDPC like codes [8] and [9]). Distributed compressed sensing for sources with memory has been proposed in [10] and [11]. It builds on the principles of standard compressed sensing [12] and joint sparsity of the signals is utilized for efficient compression. However, the effect of source quantization has not been fully addressed, specifically for the case of coarse quantization of sensor sources where the quantized signals may not be sparse in any basis [10]. The other DSC research direction builds directly on source coding methodologies. Algorithms for distributed vector quantizer design have been proposed in [13]–[15] with major or exclusive focus on memoryless sources. The source coding perspective will be most relevant to us here. The temporal correlations within a source can be exploited by blocking sources into large vectors, but such a scheme will have high complexity and will be extremely sensitive to initialization and poor local optima [16]–[18]. Motivated by these observations, a notable approach to predictive coding of correlated sources has been proposed in [17] where a uniform quantization grid was imposed on the product space (across sources) of prediction errors, on which the main support of the joint distribution was identified and a DSC code devised. The emphasis in that paper's results was on the design of optimal predictor filters in such distributed setting and on how they deviate from the case of non-distributed predictive coding. Also in [19], an algorithm for predictive coding of correlated sources exhibiting high intersource correlation was given where different components (encoder and decoders) were designed. However in both the previous settings, neither the optimality of the algorithms was proven nor the system can be guaranteed to be drift-free for all values of intersource/temporal correlations. We have proposed optimal algorithms with "zero-drift" and "controlled-drift" for distributed predictive coding in [20] and [21]. The "controlled-drift" algorithm includes the zero-drift approach as a special case that emerges whenever the impact of potential drift overwhelms the benefits of improved prediction. Both the DPC schemes also subsume as special extreme cases a) separate predictive coding of sources and b) memoryless distributed coding.

Another design difficulty whose origins are in standard predictive quantizer design [22] is exacerbated in the distributed setting. On the one hand, open-loop design is simple and stable but the quantizer is mismatched with the true prediction error statistics (as the system eventually operates in closed loop). On the other hand, if a distributed quantizer is designed in closed loop, the effects of quantizer modifications are unpredictable as quantization errors are fed back through the prediction loop and can build up. Hence, the procedure is unstable and may not converge. The effect is greatly exacerbated in the case of DPC. This will be explained in more details in Sections II-B-4 and II-C. To circumvent these difficulties, we have used the technique of asymptotic closed-loop (ACL) design [23], [24] which we rederive for DPC system design. Within the ACL framework, the design is effectively in open loop within iterations (eliminating issues of error buildup through the prediction loop), while ensuring that asymptotically, the prediction error statistics converge to closed-loop statistics. In other words, the prediction loop is essentially closed asymptotically.

The organization of the rest of the paper is as follows. In Section II, we state the problem formally, introduce notation, specify the components of the DPC system and highlight the need of ACL approach for DPC design. Section III describes the zero-drift scheme while Section IV motivates and specifies the components in controlled-drift based DPC design. Simulation results are summarized in Section V followed by conclusions in Section VI.

II. Distributed Predictive Coding

A. Problem Statement and Preliminaries

Consider the simplest distributed source coding scenario of Fig. 1. For brevity, we restrict the presentation to two sources (generalization to an arbitrary number of sources is straightforward). Here X and Y are two continuous amplitude, correlated (scalar or vector) sources with memory. The two source encoders compress and transmit source information at rates R_1 and R_2 bits per source sample respectively, to the central unit (joint decoder). The objective is to minimize the following expected distortion cost:

$$D = E\left\{\alpha d(X,\hat{X}) + (1-\alpha) d(Y,\hat{Y})\right\} \tag{1}$$

where $d(\cdot,\cdot)$ is an appropriately defined distortion measure, $\hat{X}$ and $\hat{Y}$ are the reconstruction values for X and Y respectively, and $\alpha \in [0,1]$ is a weighting factor that accounts for the relative importance of the sources at the decoder.

We further assume that predictive coding is employed to exploit temporal redundancies (we will restrict the scope to linear prediction). The prediction errors e_x (for X) and e_y (for Y) will be correlated. Therefore, instead of the standard predictive quantizer, a distributed quantizer needs to be designed to exploit intersource correlations. A mechanism to enable full leveraging of information from another correlated source requires that the encoder and decoder reconstruction of the prediction errors differ. We begin by describing the "zero-drift" approach wherein both the source encoder and decoder have access to exactly the same prediction error reconstruction for the prediction loop and then propose a "controlled drift" approach where the constraint of zero-drift is relaxed.

B. Zero Drift Approach

1) Encoder: The zero-drift distributed predictive encoder for source X is depicted in Fig. 2. The input to the high resolution quantizer Q_x is $e_x = X - \tilde{X}_{\text{enc}}$ where $\tilde{X}_{\text{enc}}$ is the predicted value of X at the encoder. Q_x maps the prediction error e_x to an index k representing Voronoi region C_k^x (a prototype can be associated with each Voronoi region). Next, a *lossy* (many to one) mapping which we refer to as the Wyner–Ziv (WZ) mapping is employed (the name loosely accounts for the fact that the scenario involves lossy coding with side information whose asymptotic performance bound was given in [2]). The WZ mapping block takes in k and outputs index $i = v(k)$ for transmission over the communication channel, and which represents region $R_i^x = \bigcup_{k:v(k)=i} C_k^x$. The *encoder codebook* C_{enc} produces $\hat{e}_{x,\text{enc}}$, the prediction error reconstruction value. An example of

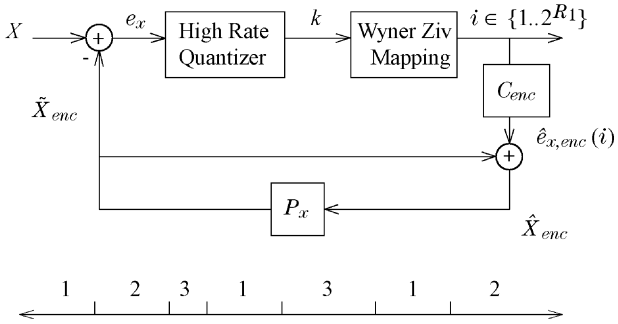

Fig. 2. Block diagram of a DPC zero-drift encoder and a scalar example of WZ mapping from prototypes (Voronoi regions) to indices.

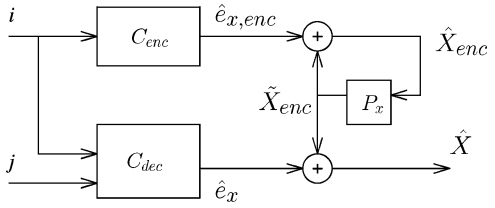

Fig. 3. DPC zero-drift decoder for source X.

WZ mapping for a scalar source with $\mathcal{K} = 7$ and $\mathcal{I} = 3$, is given in Fig. 2.

Note that the role of high-resolution quantizer is only to discretize the source signal (e_x). This can be considered as a fine quantizer and the WZ mapping block as the corresponding coarse quantizer. The WZ mapping module exploits the inter-source correlation by simply combining different regions of the source (e_x in this case) to the same index in order to reduce the transmission rate. For example, in Fig. 2, the second and seventh regions are mapped to the same index $i = 2$. At the decoder, using information from the other correlated source these regions can be distinguished.

The reconstructed residual $\hat{e}_{x,\mathrm{enc}}$ is added to $\tilde{X}_{\mathrm{enc}}$ to obtain $\hat{X}_{\mathrm{enc}}$, the sample reconstruction value for the encoder prediction loop. A linear predictor P_x is applied to $\hat{X}_{\mathrm{enc}}$ to predict the next source sample. For Y, we similarly define the quantizer Q_y, regions C_l^y and R_j^y. Here, the L Voronoi regions are mapped to J indexes via a WZ mapping $w(l) = j$. Next, we explain the functioning of the distributed predictive decoder in the zero-drift setting.

2) Decoder: The decoder module in charge of reproducing X (see Fig. 3) receives indexes i and j from sources X and Y respectively. Index i is first used to reconstruct $\hat{e}_{x,\mathrm{enc}}$ so that the encoder prediction loop can be exactly replicated without error or potential drift to generate $\hat{X}_{\mathrm{enc}}$ and $\tilde{X}_{\mathrm{enc}}$ via the predictor P_x. Given the index pair (i, j), the decoder retrieves $\hat{e}_x$ from the *decoder codebook*, C_{dec}, and adds it to $\tilde{X}_{\mathrm{enc}}$ to obtain the decoder reconstruction $\hat{X}$.

3) Observations and Intuitive Considerations: It is important to note that the Wyner–Ziv mappings compromise the quality of the sample reconstruction in the prediction loop in order to exploit intersource correlation and improve the decoder reconstruction. In particular, region $R_i^x = \bigcup_{k:v(k)=i} C_k^x$ is typically formed as a union of distant Voronoi cells C_k^x in the hope that the information from source Y will allow the decoder to separate them (see the example mapping in Fig. 2). A fundamental tradeoff emerges, as in order to exploit intersource correlations between e_x and e_y to better reconstruct the current sample at the decoder, we compromise the performance of the prediction loop and hence the quality of future reconstruction.

We should also re-emphasize that $\hat{X}_{\mathrm{enc}}$ is a (coarse) reconstruction of X which only serves the prediction loop, and is generally different from $\hat{X}$, the decoder reconstruction of X. Also note that the "encoder codebook" C_{enc} which is used in the prediction loop at both the encoder and the decoder is, in general, different from the "decoder codebook" C_{dec} (used only at the decoder).

4) Naive Approach for DPC Design: One can argue that predictive coding per se is largely a solved problem and a predictive quantizer module can be straightforwardly integrated with existing distributed memoryless coding methodologies (such as in [15]) to obtain a DPC system. The idea in such a naive approach will be to first obtain a set of prediction error residuals (e_x, e_y). Let us assume that these are initialized with the open-loop prediction errors. High rate quantizers are designed based on these error statistics using a standard quantizer design algorithm [25] (Ideally this design should be performed jointly with other system components, but additional gains due to joint design are expected to be very nominal. We therefore assume that these high rate quantizers are directly designed independently based on the error statistics). Then a distributed coder will be designed to minimize the following distortion cost between the prediction errors

$$E\left[\alpha d(e_x, \hat{e}_x) + (1-\alpha) d(e_y, \hat{e}_y)\right] \tag{2}$$

(see, e.g., DSC in [15]) (similar to the practice for traditional single-source predictive quantizer, wherein the quantizer is designed to minimize the distortion between prediction error and its reconstruction). This will resemble the open-loop design in traditional single-source predictive coding. Note that in the standard distributed coder algorithm, only the WZ mappings and decoder codebooks will be designed. For the subsequent iterations (of closed- loop predictive quantizer design) for source X, $\hat{e}_{x,\mathrm{enc}}$ will be calculated solely based on index i, since this is the only common information guaranteed to be available at both the encoder and decoder (index j from source Y is available only at decoder). Next one computes $\hat{e}_{x,\mathrm{enc}}(i)$, corresponding to transmitted index i as $\hat{e}_{x,\mathrm{enc}}(i) = E(e_x | e_x \in R_i^x)$. In other words, similar to the practice in quantizer design, the encoder codebook simply calculates $\hat{e}_{x,\mathrm{enc}}(i)$ as the centroid of region R_i^x (we assume squared error distortion measure for simplicity). Using this, the sequences $\hat{X}_{\mathrm{enc}}$, $\tilde{X}_{\mathrm{enc}}$ and prediction errors e_x will be computed in closed loop. The crucial point to note in such a design is that $\hat{e}_{x,\mathrm{enc}}(i)$ for index i is a very coarse estimate for e_x. For example, in Fig. 2, the high resolution quantizer divides the source (e_x) space into seven separate regions. Now for index $i = 2$, $\hat{e}_{x,\mathrm{enc}}$ calculated as above will lie somewhere in regions in the middle of the line. This will cause the estimate $\hat{X}_{\mathrm{enc}}$ to be coarse as well and degrade the performance of the prediction loop. The prediction error statistics for subsequent samples will differ greatly from those assumed during the distributed coder design and may even cause instability as will be illustrated in the results section. This shortcoming is primarily due to neglecting the impact of the feedback prediction loop during the design of the distributed coder. Hence, there is a major conflict between

the objectives of distributed quantization and predictive coding, and the corresponding tradeoff should be explicitly optimized.

5) Closed Loop Versus ACL Design: For conceptual simplicity, let us consider first order linear prediction. We note that the quantized error sample $\hat{e}_{x,\text{enc}}$ at time n impacts the sequence $\tilde{X}_{\text{enc}}$ and $\hat{X}$ from time $n+1$ onwards due to the presence of the prediction loop. On the other hand, $\hat{e}_x$ at time n only impacts the current $\hat{X}$ (at time n), as is explicitly depicted in Fig. 3. Hence, if one tries to directly design a *distributed quantizer* for the quantities being quantized, namely, the pair of prediction errors $\{e_x, e_y\}$ to minimize the distortion in (2) (as was done in the naive approach), the role of prediction loop (affect of $\hat{e}_{x,\text{enc}}$) on source reconstruction $\hat{X}$ is neglected. This actually implies that the ultimate end-to-end distortion in (1) will not be minimized.

However, if the DPC decoder were to perform in "open loop" as shown in Fig. 4(b), then a particular sample of $\hat{e}_{x,\text{enc}}$ will affect only the next sample (in case of m^{th} order linear predictor, it will affect m future samples) of $\hat{X}$ and not all the samples following it. Thus, if we can allow both $\hat{e}_{x,\text{enc}}$ (through the prediction loop) and $\hat{e}_x$ impact $\hat{X}$ simultaneously and not separate distributed and predictive coding as was being done in naive approach, the ultimate end-to-end distortion in (1) will be minimized. This is our main rationale of adopting the asymptotic closed-loop (ACL) approach [23], [24] for DPC system design, in which the design iterations are performed in open loop and the prediction loop is essentially closed asymptotically. The functioning of the ACL based DPC decoder will be explained in detail in Section III. We also show DPC encoder in Fig. 4(a). An important characteristic of the ACL technique is that the design is performed in open loop but as the algorithm converges, the prediction loop is effectively closed and the operation mimics closed loop. In the next subsection, we briefly explain the ACL approach for predictive quantizer design for a single source and how ACL can be adapted for zero-drift DPC design.

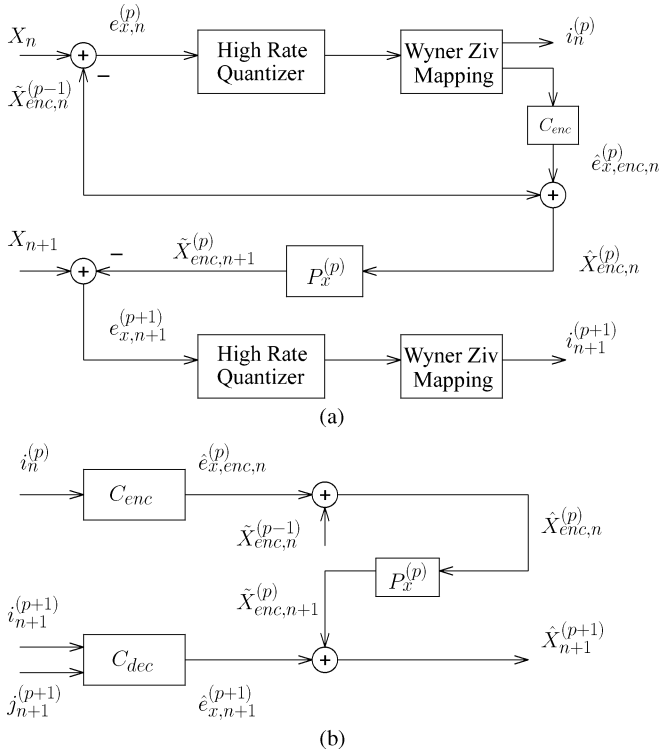

Fig. 4. DPC zero-drift encoder and decoder for source X during the design phase. Note that the prediction is actually performed in open loop. (a) Encoder; (b) decoder.

C. Asymptotic Closed Loop for Predictive Quantization (Review)

A predictive quantizer can be designed using an open-loop (OL) or closed-loop (CL) approach [22]. In OL a training set of prediction errors is generated from the original sequence of samples and is independent of the quantizer. A greedy design algorithm (e.g., Lloyd's [25]) is therefore stable and converges to a local minimum. However, the prediction loop must be closed to operate the designed system, resulting in prediction-error statistics that differ from those observed during design. Hence, the system performance is suboptimal. In CL, the system iterates in closed loop to generate new training data, for redesign of the quantizer, until (hopefully) convergence. However, since the training set changes with each iteration, each redesigned quantizer is applied to error statistics it had not been designed for. Moreover, the change in statistics is generally unpredictable as, due to the prediction loop that feeds back errors, there can be distortion build up as the sequence is processed causing non-stationary statistics and actual divergence (in terms of the performance cost). In general, there is no guarantee that the algorithm will converge and the procedure may be unstable [22].

The asymptotic closed-loop (ACL) design approach [23], [24] mitigates these shortcomings of traditional predictive coder design. A subterfuge is employed wherein the design is effectively performed in open loop, where each quantizer is designed for the statistics of the exact signal it then quantizes to produce a new sequence of reconstruction for the next iteration, thereby circumventing stability issues. Asymptotically, the loop is virtually closed in the sense that the design approaches closed-loop statistics despite open-loop operation within each iteration.

More specifically, for a given quantizer $Q^{(p-1)}$ and reconstruction sequence $\hat{X}^{(p-1)}$ obtained at iteration $p-1$, a new training set of prediction errors $T^{(p)} = \{e^{(p)}_n\}_{n=1}^{N}$ for the p^{th} iteration is generated as:

$$e^{(p)}_n = x_n - P\left[\hat{x}^{(p-1)}_{n-1}\right] \tag{3}$$

where the subscript n denotes time and P is the predictor. Using $T^{(p)}$, a new quantizer $Q^{(p)}$ is designed and a new set of reconstruction values for X is obtained by applying the new quantizer on $T^{(p)}$ itself as:

$$\hat{x}^{(p)}_n = P\left[\hat{x}^{(p-1)}_{n-1}\right] + Q^{(p)}\left[e^{(p)}_n\right]. \tag{4}$$

It should be noted that the prediction is not from the preceding sample reconstruction at the current iteration, but rather from the *fixed* reconstruction sequence of the previous iteration. Hence, unlike CL, the prediction errors to be quantized are fixed and do not change as we modify the quantizer. Since the quantizer is applied to the exact error training set for which it was designed, it is the best quantizer for the job and hence the distortion cost will

decrease. This will result in better prediction. A new prediction error training set $T^{(p+1)}$ is then obtained and the procedure is performed until a convergence criterion is met. Since the entire design is performed in open loop, it is stable. At convergence, the quantizer updates are vanishingly small $Q^{(p+1)} \approx Q^{(p)}$. Therefore, the reconstructed sequence is unchanged with iterations, i.e., $\hat{x}_n^{(p+1)} \approx \hat{x}_n^{(p)}$ implying $P[\hat{x}_{n-1}^{(p+1)}] \approx P[\hat{x}_{n-1}^{(p)}]$ which means that asymptotically we are effectively predicting from the previous sample reconstruction in the current iteration, i.e., the loop is effectively closed. So, even though the algorithm is always running in open loop, the design asymptotically approaches closed-loop conditions. More details about ACL are given in [23] and [24].

III. ACL for Zero-Drift Distributed Predictive Coding

The ACL distributed predictive decoder (zero-drift approach) for source X is shown in Fig. 4. Here $i_n^{(p)}$, $j_n^{(p)}$ denote the received indexes in the p^{th} ACL iteration. $e_{x,\text{enc},n}^{(p)}$ is the prediction error estimate of the encoder codebook during iteration p for n^{th} time sample. The other entities $\tilde{X}_{\text{enc},n}^{(p-1)}$, $\hat{X}_{\text{enc},n}^{(p)}$, etc., are correspondingly defined. During the design iteration, the prediction loop is open as shown. The distortion cost to be minimized is:

$$D^{(p)} = E\left[\alpha d\left(X, \hat{X}^{(p+1)}\right) + (1-\alpha) d\left(Y, \hat{Y}^{(p+1)}\right)\right]. \quad (5)$$

Note that during iteration p, we seek to minimize the ultimate cost at iteration $p+1$. Asymptotically, this makes no difference. This setting is used to ensure that the direct impact of the present error reconstruction ($\hat{e}_{x,n+1}^{(p+1)}$), and previous error reconstruction ($\hat{e}_{x,\text{enc},n}^{(p)}$) via the prediction loop on $\hat{X}_{n+1}^{(p+1)}$ is taken into account for effective update rules. Also, since the design is actually in open loop, $\hat{e}_{x,\text{enc},n}^{(p)}$ affects $\hat{X}^{(p+1)}$ at time $n+1$ only.

A. Update Rules: Zero-Drift DPC

For simplicity, we assume that $d(\cdot,\cdot)$ is the squared error distortion measure. The decoder codebook, encoder codebook, Wyner–Ziv mappings and the predictor are updated iteratively using the following steps:

1) **Decoder Codebook** (C_{dec}): Entry (i,j), $i = 1:\mathcal{I}$ and $j = 1:\mathcal{J}$ is given by (6), shown at the bottom of the page. Here $\{\hat{e}_x(i,j)\}^{(p)}$ and $\{\hat{e}_x(i,j)\}^{(p+1)}$ denote the decoder codebook entry (i,j) for ACL iteration p and $p+1$ respectively.
2) **Encoder Codebook** (C_{enc}): Entry i, $i = 1:\mathcal{I}$ is given by (7), shown at the bottom of the page, where the resulting prediction error of source X depends on ψ via $e_{x,n+1}^{(p+1)} = x_{n+1} - P_x^{(p)}[\tilde{x}_{\text{enc},n}^{(p-1)} + \psi]$. Note that $\hat{e}_{x,n+1}^{(p+1)}$ is shorthand for $\hat{e}_x(i_{n+1}^{(p+1)}, j_{n+1}^{(p+1)})$, the reconstructed value of $e_{x,n+1}^{(p+1)}$. Further $\{\hat{e}_{x,\text{enc}}(i)\}^{(p)}$ and $\{\hat{e}_{x,\text{enc}}(i)\}^{(p+1)}$ respectively denote the encoder codebook entry i during ACL iteration p and $p+1$.
3) **WZ Mappings**: For $k = 1,\ldots,K$, assign k to index $i = \{v(k)\}^{(p)}$ such that (8), shown at the bottom of the page. Here $\{v(k)\}^{(p)}$ and $\{v(k)\}^{(p+1)}$ denote the WZ mapping for X during ACL iteration p and $p+1$ respectively. Note that the two subsets of source X samples corresponding to prediction errors in iterations p and $p+1$ (i.e., $n : e_{x,n}^{(p)} \in C_k$ and $n : e_{x,n+1}^{(p+1)} \in C_k$) may overlap. We need to account for the effect of each element in either subset, hence we consider the union of these subsets via an "OR" condition in the summand of (8).
4) **Predictor**: See Section III-B.

The corresponding update rules for source Y are similarly obtained. Note that i and j point to codebook entries, subscript n indicates time, and superscript p indicates the ACL iteration. To reduce clutter, superscripts were omitted where obvious, e.g., R_i for R_i^x.

$$\{\hat{e}_x(i,j)\}^{(p)} = \{\hat{e}_x(i,j)\}^{(p+1)} = \arg\min_{\phi} \sum_{n:\left(e_{x,n}^{(p+1)}, e_{y,n}^{(p+1)}\right) \in R_i \times R_j} d\left(e_{x,n}^{(p+1)}, \phi\right). \quad (6)$$

$$\{\hat{e}_{x,\text{enc}}(i)\}^{(p)} = \{\hat{e}_{x,\text{enc}}(i)\}^{(p+1)} = \arg\min_{\psi} \sum_{n:e_{x,n}^{(p)} \in R_i} \left[\alpha d\left(e_{x,n+1}^{(p+1)}, \hat{e}_{x,n+1}^{(p+1)}\right) + (1-\alpha) d\left(e_{y,n+1}^{(p+1)}, \hat{e}_{y,n+1}^{(p+1)}\right)\right] \quad (7)$$

$$\begin{aligned}\{v(k)\}^{(p)} = \{v(k)\}^{(p+1)} = \arg\min_{i\in\{1\ldots I\}} & \sum_{n:e_{x,n}^{(p)} \in C_k \text{ or } e_{x,n+1}^{(p+1)} \in C_k} \alpha d\left(x_{n+1}, P_x^{(p)}\left[\tilde{x}_{\text{enc},n}^{(p-1)} + \hat{e}_{x,\text{enc}}(i)\right] + \hat{e}_x\left(i, j_{n+1}^{(p+1)}\right)\right) \\ & + (1-\alpha) d\left(y_{n+1}, P_y^{(p)}\left[\tilde{y}_{\text{enc},n}^{(p-1)} + \hat{e}_{y,\text{enc}}\left(j_n^{(p)}\right)\right] + \hat{e}_y\left(i, j_{n+1}^{(p+1)}\right)\right). \end{aligned} \quad (8)$$

In order to account for the impact of adjustments to, e.g., the decoder codebook, the update rule must involve values indexed by $p+1$; see Fig. 4. We hence increment the iteration counter $p \leftarrow p+2$ at the end of each ACL iteration. This automatically implies that the encoder and decoder codebooks, and WZ mapping entries, are unchanged between p and $p+1$, as is made explicit in the update equations. Effectively, we use the ACL counter sequence $p = 1, 3, 5, \ldots$. Note that this subterfuge to enable proper update rules makes no asymptotic difference.

B. Predictor Optimization

To obtain an effective update rule for the predictor, we keep the various codebooks and WZ mappings fixed, and set to zero the partial derivative of the distortion cost in (5) with respect to the predictor. For the fixed set of reconstructed sequence $\{\hat{x}_{\text{enc},n}^{(p-1)}\}_{n=0}^{N}$, the prediction error at iteration p is calculated as

$$e_{x,n}^{(p)} = x_n - P_x^{(p)} \left[\hat{x}_{\text{enc},n-1}^{(p-1)}\right]. \tag{9}$$

For notational simplicity, we break the distortion cost in (5) as $D^{(p)} = \alpha D_x^{(p)} + (1-\alpha) D_y^{(p)}$, where $D_x^{(p)}$ and $D_y^{(p)}$ are the contributions to the distortion from sources X and Y, respectively: $D_x^{(p)} = E[d(X, \hat{X}^{(p+1)})]$ and $D_y^{(p)} = E[d(Y, \hat{Y}^{(p+1)})]$. The term $D_x^{(p)}$ can be rewritten as follows:

$$D_x^{(p)} = \frac{1}{N} \sum_{n=1}^{N} \left[X_n - \hat{X}_n^{(p+1)}\right]^2 \tag{10}$$

$$= \frac{1}{N} \sum_{n=1}^{N} \left[X_n - \hat{e}_{x,n}^{(p+1)} - \tilde{X}_{\text{enc},n}^{(p)}\right]^2 \tag{11}$$

$$= \frac{1}{N} \sum_{n=1}^{N} \left[X_n - \hat{e}_{x,n}^{(p+1)} - P_x^{(p)} \hat{X}_{\text{enc},n-1}^{(p)}\right]^2 \tag{12}$$

$$= \frac{1}{N} \sum_{n=1}^{N} \left[X_n - \hat{e}_{x,n}^{(p+1)} - P_x^{(p)} \left(\hat{e}_{x,\text{enc},n-1}^{(p)} + \tilde{X}_{\text{enc},n-1}^{(p-1)}\right)\right]^2 \tag{13}$$

where we replaced expectation with sample averaging over N samples in the training set.

While minimizing the distortion cost $D^{(p)}$ in (5) with respect to $P_x^{(p)}$, we neglect the effect of adjusting predictor $P_x^{(p)}$ on the reconstructed prediction error $\hat{e}_{x,n}^{(p+1)}$, which is effectively a coarse quantizer output (implemented by a high resolution quantizer followed by WZ mapping). This approximation is standard practice in predictor optimization in traditional (single-source) predictive quantizer design [22], [26].

Setting $\nabla_{P_x^{(p)}} D^{(p)} = \nabla_{P_x^{(p)}} D_x^{(p)} = 0$, we obtain the matrix equation

$$\frac{1}{N} \sum_{n=1}^{N} \left[X_n - \hat{e}_{x,n}^{(p+1)} - P_x^{(p)} \left(\hat{e}_{x,\text{enc},n-1}^{(p)} + \tilde{X}_{\text{enc},n-1}^{(p-1)}\right)\right] \cdot \left[\hat{e}_{x,\text{enc},n-1}^{(p)} + \tilde{X}_{\text{enc},n-1}^{(p-1)}\right]^T = 0 \tag{14}$$

where superscript T denotes matrix transposition. The solution is

$$P_x^{(p)} = A_x^{(p)} \left(B_x^{(p)}\right)^{-1} \tag{15}$$

where

$$A_x^{(p)} = \sum_{n=1}^{N} \left(X_n - \hat{e}_{x,n}^{(p+1)}\right) \left(\hat{e}_{x,\text{enc},n-1}^{(p)} + \tilde{X}_{\text{enc},n-1}^{(p-1)}\right)^T \tag{16}$$

and

$$B_x^{(p)} = \sum_{n=1}^{N} \left(\hat{e}_{x,\text{enc},n-1}^{(p)} + \tilde{X}_{\text{enc},n-1}^{(p-1)}\right) \left(\hat{e}_{x,\text{enc},n-1}^{(p)} + \tilde{X}_{\text{enc},n-1}^{(p-1)}\right)^T. \tag{17}$$

C. Algorithm Description

In ACL iteration p, the various codebooks, WZ mappings and the predictors are updated iteratively. Convergence is guaranteed within an ACL iteration since the update steps (each consisting of optimizing one module while fixing the others) are monotone non-increasing in the distortion (5).

For the next ACL iteration, the sequence $\hat{e}_{x,\text{enc}}^{(p+1)}$ is calculated using the index sequence $i^{(p+1)}$. The reconstructed sequence $\hat{X}_{\text{enc}}^{(p+1)}$ is obtained as

$$\hat{X}_{\text{enc}}^{(p+1)} = \tilde{X}_{\text{enc}}^{(p)} + \hat{e}_{x,\text{enc}}^{(p+1)} \tag{18}$$

and kept fixed for ACL iteration $p+2$. Recall that since the update rules involve parameters in iteration p and $p+1$, we increment the iteration counter $p \leftarrow p+2$. Next the predicted sequence is computed as

$$\tilde{X}_{\text{enc}}^{(p+1)} = P_x^{(p)} \hat{X}_{\text{enc}}^{(p+1)}. \tag{19}$$

We initialize the predictor for the ACL iteration $p+2$ as $P_x^{(p+2)} = P_x^{(p)}$ and proceed to the next ACL iteration. A flowchart describing the algorithm is given in Fig. 5.

IV. Controlled Drift Approach

A. Motivation and Description

In the zero drift approach, to avoid any potential mismatch the encoder codebook for source X (see Figs. 2 and 4) was restricted to have index i as input. However, the source encoder for X has complete knowledge of the prediction error (e_x) itself or effectively the index k (which is the output of high resolution quantizer used primarily to discretize the source), while the decoder has additional knowledge about the prediction error from the correlated source Y, in the form of index j. This implies that there exist some (elusive) additional information that could be exploited, if an appropriate means were devised. This may be done by using different codebooks for the prediction loop at the decoder and encoder, specifically assigning k as the input to the encoder codebook, while the decoder *loop* codebook has i and j as inputs. This flexibility enables better exploitation of inter-source correlation, at the cost of some drift in the system. However, appropriate design of encoder and loop codebooks will optimize the precise overall performance while accounting for and managing the drift. Note that the controlled-drift approach actually subsumes the zero-drift scheme as an extreme special case where the encoder and loop codebooks are effectively the same and depend only on i. The encoder and decoder employed for the controlled drift approach during system operation are depicted in Fig. 6 and Fig. 7. However, during the ACL design,

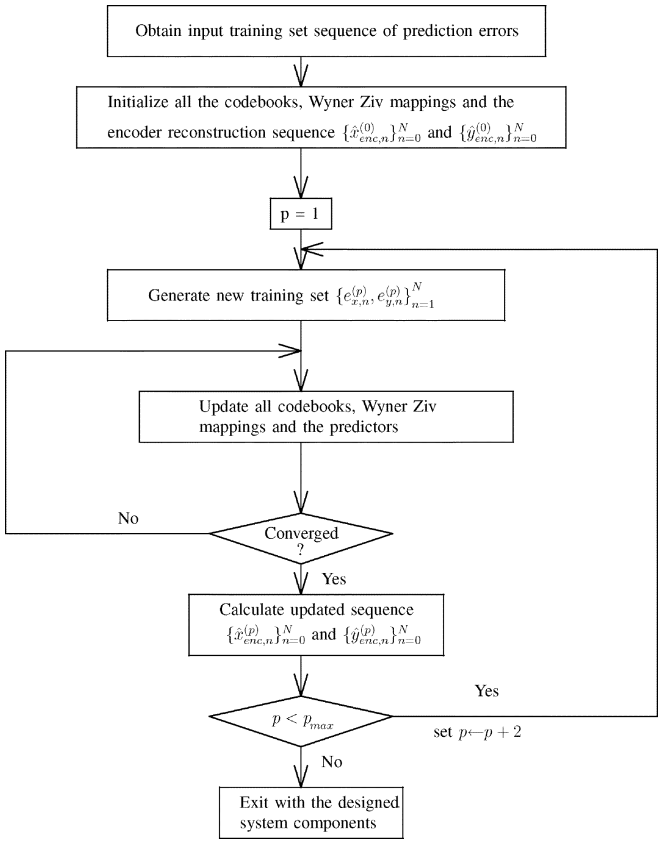

Fig. 5. Flowchart of asymptotic closed-loop design procedure for distributed predictive coding.

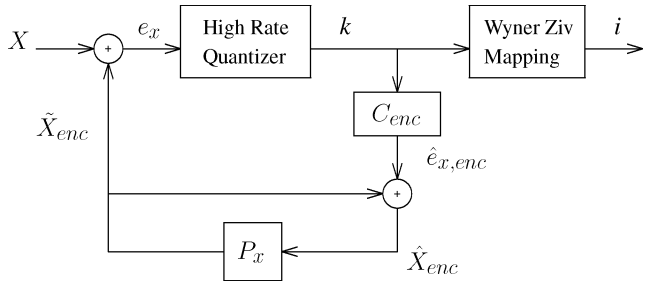

Fig. 6. Controlled-drift DPC encoder.

the prediction loop is open as shown for the decoder in Fig. 8. We next specify the update rules for controlled drift DPC which parallel those of zero-drift DPC.

B. Update Rules: Controlled Drift DPC

As earlier, we assume mean squared error distortion for simplicity. The notation in what follows is necessarily heavy

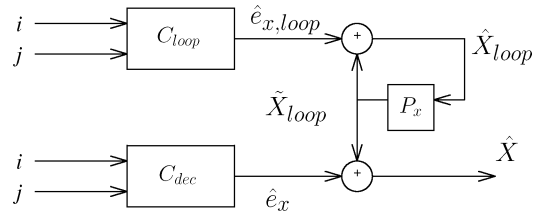

Fig. 7. Controlled-drift DPC decoder.

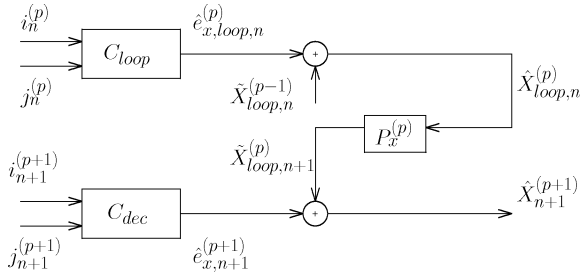

Fig. 8. Controlled-drift DPC decoder during design phase.

due to the multiple indexing involved; in a nutshell we alternate between optimization of the decoder codebook, encoder codebook, loop codebook, WZ mapping and predictor. The update rules below are specified in terms of the subset of distortion terms that depend on the parameters being updated; while avoiding overly detailed notation. As mentioned earlier in the zero-drift approach, the various codebooks and WZ mapping corresponding to p and $p+1$ are same and we increment the ACL iteration counter as $p \leftarrow p+2$.

1) **Decoder Codebook** (C_{dec}): Entry (i,j), $i = 1:\mathcal{I}$ and $j = 1:\mathcal{J}$ is obtained as

$$\begin{aligned}&\{\hat{e}_x(i,j)\}^{(p)}\\&\quad=\{\hat{e}_x(i,j)\}^{(p+1)}\\&\quad=\arg\min_{\phi}\sum_{n:\left(e^{(p+1)}_{x,n+1},e^{(p+1)}_{y,n+1}\right)\in R_i\times R_j} d\Big(x_{n+1},\tilde{x}^{(p)}_{\text{loop},n+1}+\phi\Big).\end{aligned} \tag{20}$$

2) **Loop Codebook** (C_{loop}): Entry (i,j), $i = 1:\mathcal{I}$ and $j = 1:\mathcal{J}$ is obtained as (21), shown at the bottom of the page, where $\hat{e}^{(p+1)}_{x,n+1}$ is shorthand notation for $\hat{e}_x(i^{(p+1)}_{n+1}, j^{(p+1)}_{n+1})$.
3) **Encoder Codebook** (C_{enc}): Entry k, $k = 1:\mathcal{K}$ is obtained as (22), shown at the bottom of the next page, where the resulting prediction error of source at encoder X depends on ζ via $e^{(p+1)}_{x,n+1} = x_{n+1} - P_x[\tilde{x}^{(p-1)}_{\text{enc},n} + \zeta]$, and $\hat{e}^{(p+1)}_{x,n+1}$, $\hat{e}^{(p+1)}_{y,n+1}$ are the reconstructed value of $e^{(p+1)}_{x,n+1}$ and $e^{(p+1)}_{y,n+1}$, respectively.

$$\{\hat{e}_{x,\text{loop}}(i,j)\}^{(p)} = \{\hat{e}_{x,\text{loop}}(i,j)\}^{(p+1)} = \arg\min_{\psi}\sum_{n:\left(e^{(p)}_{x,n},e^{(p)}_{y,n}\right)\in R_i\times R_j} d\left(x_{n+1}, P_x\left(\tilde{x}^{(p-1)}_{\text{loop},n}+\psi\right)+\hat{e}^{(p+1)}_{x,n+1}\right) \tag{21}$$

4) **WZ Mappings**: For $k = 1 : \mathcal{K}$, assign k to index $i = \{v(k)\}^{(p)}$ such that [see (23), shown at the bottom of the page].
5) **Predictor**: See the predictor update in zero-drift DPC approach.

To reduce clutter, we have again omitted the superscripts where obvious, e.g., R_i rather than R_i^x etc. We optimize the predictor for both the zero-drift and controlled-drift DPC schemes using the update rules derived in Section III-B. However, one may still do better in the case of controlled-drift scheme by allowing different prediction filters at the encoder and decoder. In our experiments, we observed that adjusting the prediction filters yielded modest performance gains and thus we skip here the derivation of optimal prediction filters in the controlled-drift setting.

V. Simulation Results

A. Performance Evaluation

The following Gauss–Markov source model is used for simulations:

$$X_n = \beta X_{n-1} + w_n \quad \text{and} \quad Y_n = \gamma Y_{n-1} + u_n \tag{24}$$

where w_n, u_n are i.i.d., zero-mean, unit variance, jointly Gaussian scalar sources with correlation coefficient ρ. A training set of size 5000 scalars is generated. We use mean squared error distortion for all simulations. The predictors P_x and P_y are first-order linear predictors. Simulation results are depicted in Fig. 9. In all simulations, the weighting coefficient of (1) is set to $\alpha = 0.5$ so that equal importance is given to both sources at the decoder. The number of prototypes is 60 for each source.

In the first experiment, $\beta = \gamma = 0.8$ and $\rho = 0.97$. Both sources are encoded at the same rate. The weighted distortion at the decoder is plotted versus the number of transmitted bits for each source. We compare a) "non-distributed" predictive coding, i.e., each source is compressed independently using standard predictive coding; b) memoryless distributed coding, i.e., no prediction is performed and a simple distributed source coder to exploit intersource correlation (this is simply designing a distributed quantizer for the source pair (X, Y) using a standard distributed quantizer algorithm, see, e.g., [15] or [18]); c) zero-drift distributed predictive coding (DPC-ZD); and d) controlled-drift distributed predictive coding (DPC-CD). The two DPC schemes (with or without drift) clearly outperform the other two compression schemes and gains of $\sim$1.7 dB are achieved (e.g., at $R_1 = R_2 = 2$ bits/sample) by the DPC-CD scheme over traditional predictive coding or memoryless distributed coding. We do not include the "naive" approach for DPC design (see Section II-B-4)) in this comparison due to severe instabilities exacerbated by the naive scheme as shown in the next subsection.

In the second experiment, $\rho = 0.96$ and the transmission rates for the sources are fixed at 2 bits/sample. The temporal correlation $\beta(= \gamma)$ is varied in this experiment. Note that the source variances change as we vary β. So we need to normalize weighted distortion by the weighted source variances. Hence, we employ the SNR defined as $(\alpha E[X^2] + (1-\alpha)E[Y^2])/(\alpha E[(X-\hat{X})^2] + (1-\alpha)E[(Y-\hat{Y})^2])$ which is a better performance metric in this experiment. We plot SNR versus temporal correlation $\beta(= \gamma)$. Note that the memoryless distributed coding scheme performs same as we vary the temporal correlation while the performance of traditional predictive coding, which exploits temporal memory, becomes better with increasing temporal correlation. The DPC schemes, which utilize both temporal and intersource correlations consistently outperform traditional predictive coding or memoryless distributed coding and gains up to 1.6 dB are achieved, e.g., at $\beta = 0.8$.

In the third experiment, $\beta = \gamma = 0.6$ and $R_1 = R_2 = 2$ bits/sample. We plot the weighted distortion versus intersource correlation ρ. The DPC-CD scheme achieves gains up to 1.1 dB (at $\rho = 0.95$) over traditional predictive coding or memoryless distributed coding. It is noteworthy that at low values of intersource correlation, the cost of a mismatch between the encoder and decoder estimates tends to overwhelm the benefits due exploiting such limited intersource correlation, and thereby compromise the overall prediction loop performance. The DPC-CD scheme thus automatically converges to a solution that mimics the zero drift scheme in such circumstances. On the other hand, at high intersource correlations whose potential benefits are considerate, the DPC-CD scheme offers additional gains over the

$$\{\hat{e}_{x,\mathrm{enc}}(k)\}^{(p)} = \{\hat{e}_{x,\mathrm{enc}}(k)\}^{(p+1)} = \arg\min_{\zeta} \sum_{n:e_{x,n}^{(p)} \in C_k} \alpha d\left(x_{n+1}, \tilde{x}_{\mathrm{loop},n+1}^{(p)} + \hat{e}_{x,n+1}^{(p+1)}\right) + (1-\alpha) d\left(y_{n+1}, \tilde{y}_{\mathrm{loop},n+1}^{(p)} + \hat{e}_{y,n+1}^{(p+1)}\right) \tag{22}$$

$$\begin{aligned}\{v(k)\}^{(p)} &= \{v(k)\}^{(p+1)} \\ &= \arg\min_{i \in \{1 \ldots I\}} \sum_{n:e_{x,n}^{(p)} \in C_k \text{ or } e_{x,n+1}^{(p+1)} \in C_k} \alpha d\left(x_{n+1}, P_x\left(\tilde{x}_{\mathrm{loop},n}^{(p-1)} + \hat{e}_{x,\mathrm{loop}}\left(i, j_n^{(p)}\right)\right) + \hat{e}_x\left(i, j_{n+1}^{(p+1)}\right)\right) \\ &\quad + (1-\alpha) d\left(y_{n+1}, P_y\left(\tilde{y}_{\mathrm{loop},n}^{(p-1)} + \hat{e}_{y,\mathrm{loop}}\left(i, j_n^{(p)}\right)\right) + \hat{e}_y\left(i, j_{n+1}^{(p+1)}\right)\right).\end{aligned} \tag{23}$$

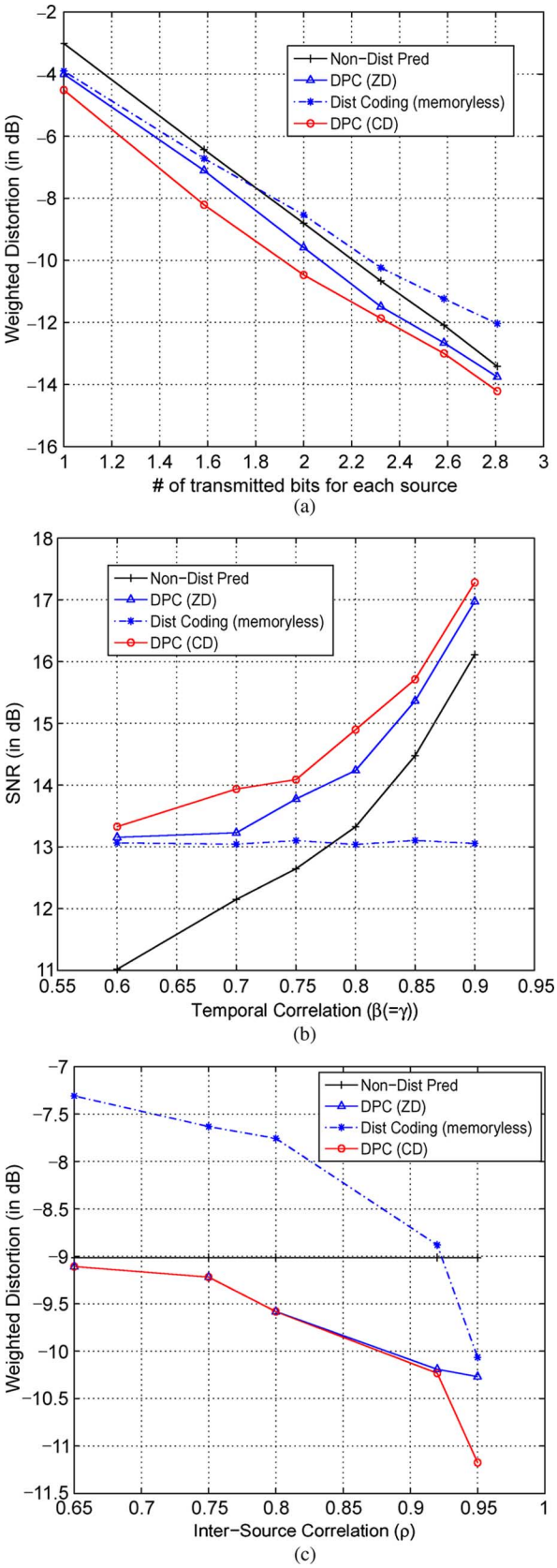

Fig. 9. Performance comparison of distributed predictive coding schemes, non-distributed predictive coding, and memoryless distributed coding. Figures (a) and (c) show weighted distortion versus rate and intersource correlation respectively. Figure (b) shows SNR versus temporal correlation. (a) Fixed intersource and temporal correlations; (b) fixed rate and intersource correlation; (c) fixed rate and temporal correlation.

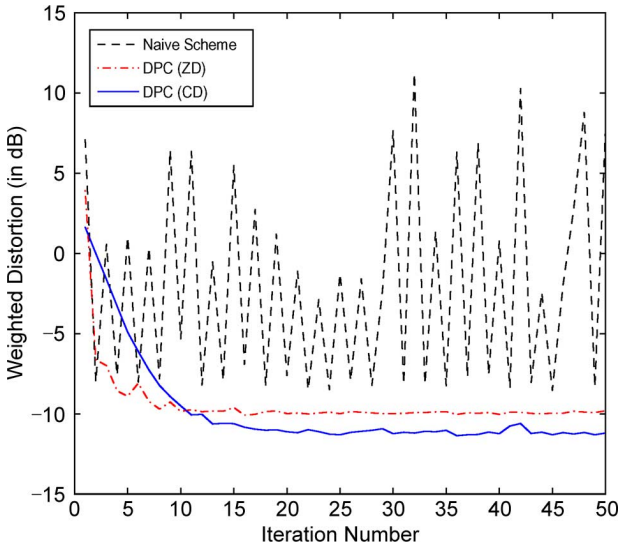

Fig. 10. Plot showing the convergence of various distributed predictive coding algorithms. Here $\rho = 0.98$, $\beta = \gamma = 0.8$, $R_1 = R_2 = 2$ bits/sample.

zero drift approach by allowing a controlled amount of mismatch between the encoder and decoder estimates and thereby exploiting intersource correlation effectively.

B. Convergence

In Fig. 10, we show the convergence (in terms of weighted distortion) of the controlled-drift and zero-drift DPC algorithms versus the number of iterations of the algorithms. The algorithms approach convergence in a small number of iterations (typically 15–20). Since DPC system design will generally be done offline and only once, the complexity should be manageable. Note that a naive combination of the distributed coding and predictive coding modules results in a highly unstable suboptimal system as was anticipated in Section II-B-4 that described the naive approach.

We observed "limit cycles" in the DPC design procedure, similar to the design of single source predictive quantizer [24]. This can be attributed to two interrelated reasons: a) during an ACL iteration, the various modules (codebooks and WZ mappings) are each greedily optimized while keeping the others fixed. This leads to convergence to a local minimum point. As we recompute the reconstruction sequences and prediction errors for the subsequent iterations, we may find different locally optimal points thereby causing the "limit cycle"; b) the update of WZ mappings (where different regions are mapped to indices) is equivalent to a complex index-assignment problem that exacerbates suboptimalities. To overcome this shortcoming of limit cycles, annealing based techniques can be employed. Note that an annealing based algorithm for single-source predictive coding via ACL was proposed in [24]. We observed in experiments that these limit cycles are small in magnitude and in general do not impact the algorithm performance.

C. Comments

We have run the various algorithms multiple times since greedy descent will converge to a local minimum, depending on initialization. It is straightforward to apply the proposed algorithms to higher vector dimensions. However, this will exacerbate the problem of local optima. We do have a proper solution to this shortcoming, which we had developed for

memoryless DSC [18], but extending and adopting such global optimization techniques for DPC is beyond the scope of this focused paper.

The main emphasis in this paper is on designing a distributed predictive coding framework where the temporal correlations within a source are exploited via a low-complexity, zero-delay predictive coding approach. Hence, we have restricted comparison to zero delay schemes only. We believe that combining our proposed distributed predictive coding framework with other schemes that exploit source memory (such as lattice vector quantization) would yield further gains. Finally, we note that the proposed methods are extendible to incorporate entropy coding.

VI. Conclusion

In this paper, we proposed iterative descent algorithms for the design of distributed predictive coding systems for spatio-temporally correlated sources. This is the typical setting for sources with memory in a sensor network. We have shown that straightforward integration of distributed coding and predictive coding modules results in a highly suboptimal system and tends to suffer from severe design instabilities. We then presented approaches, namely, zero-drift DPC and controlled drift DPC. The zero-drift approach allows no mismatch between the encoder and decoder prediction error estimates. To utilize intersource correlation more efficiently, the constraint of zero-drift is relaxed in the controlled-drift approach. Simulation results show that the two proposed distributed predictive schemes perform significantly better than memoryless distributed coding and traditional single-source predictive coding schemes. Finally the controlled-drift DPC scheme offers additional gains over the zero-drift DPC scheme, especially for high intersource correlations.

References

[1] D. Slepian and J. Wolf, "Noiseless coding of correlated information sources," *IEEE Trans. Inf. Theory*, vol. 19, no. 4, pp. 471–480, Jul. 1973.

[2] A. D. Wyner and J. Ziv, "The rate-distortion function for source coding with side-information at the decoder," *IEEE Trans. Inf. Theory*, vol. 22, pp. 1–10, Jan. 1976.

[3] S. S. Pradhan and K. Ramchandran, "Distributed source coding using syndromes (DISCUS): Design and construction," *IEEE Trans. Inf. Theory*, vol. 49, no. 3, pp. 626–643, Mar. 2003.

[4] S. S. Pradhan, J. Kusuma, and K. Ramchandran, "Distributed compression in a dense microsensor network," *IEEE Signal Process. Mag.*, vol. 19, no. 2, pp. 51–60, Mar. 2002.

[5] Z. Xiong, A. Liveris, and S. Cheng, "Distributed source coding for sensor networks," *IEEE Signal Process. Mag.*, vol. 21, no. 5, pp. 80–94, Sep. 2004.

[6] X. Wang and M. T. Orchard, "Design of trellis codes for source coding with side information at the decoder," in *IEEE Data Compression Conf.*, Mar. 2001, pp. 361–370.

[7] P. Mitran and J. Bajcsy, "Coding for the Wyner–Ziv problem with turbo-like codes," in *Proc. IEEE Int. Symp. Information Theory*, Jun. 2002, p. 91.

[8] J. Garcia-Frias and Y. Zhao, "Compression of correlated binary sources using turbo codes," *IEEE Commun. Lett.*, vol. 5, no. 10, pp. 417–419, Oct. 2001.

[9] A. D. Liveris, Z. Xiong, and C. Eorghiades, "Compression of binary sources with side information at the decoder using LDPC codes," *IEEE Commun. Lett.*, vol. 6, no. 10, pp. 440–442, Oct. 2002.

[10] D. Baron, M. Wakin, M. F. Durate, S. Sarvotham, and R. G. Baraniuk, Distributed Compressed Sensing. [Online]. Available: www.dsp.rice.edu/cs

[11] M. Duarte, M. Wakin, D. Baron, and R. Baraniuk, "Universal distributed sensing via random projections," in *Proc. IEEE Information Processing in Sensor Networks*, Apr. 2006, pp. 177–185.

[12] D. L. Donoho, "Compressed sensing," *IEEE Trans. Inf. Theory*, vol. 52, no. 4, pp. 1289–1306, Apr. 2006.

[13] J. Cardinal and G. B. Assche, "Joint entropy-constrained multiterminal quantization," in *Proc. IEEE Int. Symp. Information Theory*, Jun. 2002, p. 63.

[14] M. Fleming, Q. Zhao, and M. Effros, "Network vector quantization," *IEEE Trans. Inf. Theory*, vol. 50, no. 8, pp. 1584–1604, Aug. 2004.

[15] D. Rebollo-Monedero, R. Zhang, and B. Girod, "Design of optimal quantizers for distributed source coding," in *Proc. IEEE Data Compression Conf.*, Mar. 2003, pp. 13–22.

[16] A. Saxena, J. Nayak, and K. Rose, "A global approach to joint quantizer design for distributed coding of correlated sources," in *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing (ICASSP)*, May 2006, vol. 2, pp. 53–56.

[17] E. Tuncel, "Predictive coding of correlated sources," in *Proc. IEEE Information Theory Workshop*, Oct. 2004, pp. 111–116.

[18] A. Saxena, J. Nayak, and K. Rose, "On efficient quantizer design for robust distributed source coding," in *Proc. IEEE Data Compression Conf.*, Mar. 2006, pp. 63–72.

[19] P. Yahampath, "Predictive vector quantizer design for distributed source coding," in *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing (ICASSP)*, Apr. 2007, vol. 3, pp. 629–632.

[20] A. Saxena and K. Rose, "Distributed predictive coding for spatio-temporally correlated sources," in *Proc. IEEE Int. Symp. Information Theory*, Jun. 2007, pp. 1506–1510.

[21] A. Saxena and K. Rose, "Challenges and recent advances in distributed predictive coding," in *Proc. IEEE Information Theory Workshop*, Sep. 2007, pp. 448–453.

[22] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression*. Norwell, MA: Kluwer, 1992.

[23] H. Khalil, K. Rose, and S. L. Regunathan, "The asymptotic closed-loop approach to predictive vector quantizer design with application in video coding," *IEEE Trans. Image Process.*, vol. 10, no. 1, pp. 15–23, Jan. 2001.

[24] H. Khalil and K. Rose, "Predictive vector quantizer design using deterministic annealing," *IEEE Trans. Signal Process.*, vol. 51, no. 1, pp. 244–254, Jan. 2003.

[25] S. P. Lloyd, "Least squares quantization in PCM," *IEEE Trans. Inf. Theory*, vol. 28, no. 2, pp. 129–137, Mar. 1982.

[26] P. C. Chang and R. M. Gray, "Gradient algorithms for designing predictive vector quantizers," *IEEE Trans. Acoustics, Speech, Signal Processing*, vol. ASSP-34, no. 4, pp. 679–690, Aug. 1986.

Ankur Saxena (S'06–M'09) was born in Kanpur, India, in 1981. He received the B.Tech. degree in electrical engineering from the Indian Institute of Technology, Delhi, in 2003 and the M.S. and Ph.D. degrees in electrical and computer engineering from the University of California, Santa Barbara, in 2004 and 2008, respectively.

He has interned at the Fraunhofer Institute of X-Ray Technology, Erlangen, Germany, and NTT DoCoMo Research Labs, Palo Alto, CA, in summers 2002 and 2007, respectively. He is currently a Postdoctoral Researcher at the University of California, Santa Barbara. His research interests span source coding, image, and video compression and signal processing.

Dr. Saxena received the President Work study award during his Ph.D. studies.

Kenneth Rose (S'85–M'91–SM'01–F'03) received the Ph.D. degree from the California Institute of Technology, Pasadena, in 1991.

He then joined the Department of Electrical and Computer Engineering, University of California, Santa Barbara, where he is currently a Professor. His main research activities are in the areas of information theory and signal processing, and include rate-distortion theory, source and source-channel coding, audio and video coding and networking, pattern recognition, and nonconvex optimization. He is interested in the relations between information theory, estimation theory, and statistical physics, and their potential impact on fundamental and practical problems in diverse disciplines.

Dr. Rose was corecipient of the 1990 William R. Bennett Prize Paper Award of the IEEE Communications Society, as well as the 2004 and 2007 IEEE Signal Processing Society Best Paper Awards.